API (Eng)

Created by Galina Bocherova, last modified by Ilia Tarasov on Sep 06, 2021

API

Visit the https://vision.ctclm.com/ to learn more on the CT Visionnaire API.

Mobile API

General

Authentication:

POST /api/v1/login

Login

Headers

```
Content-Type: "application/json"
```

Body

```
{ "organizationId" : "18-digit indicator of the SF Instance ID", "token" : "CloudToken from the Cust
```

Result

```
{ "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJvcmdhbml6YXRpb25JZCI6Im9yZzU0MzIxMTIzNDU2Nzg5MC
```

In response, a JWT (JSON web token) is being received. This token must be added when addressing the other API methods.

This token can be added by:

- Adding a parameter to the URL: https://vision.ctclm.com/apimethod?access_token=jwt_token ;
- Including the token into the 'body', if JSON can be sent through;

```
{ "access_token": "jwt_token" .. other parameters }
```

- Adding it to the 'header' of the request: "x-access-token": jwt_token ;

The JWT's lifetime is 15 minutes. It can be checked on the device before submitting the request. To do so, cut the token string at the "." and decode the second part from base64 to text.

As a result, the JSON will have the following form:

Payload

```
{ "organizationId":"18 sf id", "isAdmin" : false, "iat":1558007339, "exp":1558008239 }
```

The "exp" parameter contains an encoded time till which the token can be used:

If a request is sent with an expired token, an error will be displayed:

Expired token

```
Status: 400 Bad Request error { "name": "TokenExpiredError", "message": "jwt expired", "expiredAt": "2
```

File upload (over 3Mb)

File upload is performed in the following steps:

1. Obtaining a link to upload to Amazon S3 storage.
2. File upload to the S3 storage.
3. Notification of the server on the successful upload to begin the recognition process.

Obtaining a link to upload to Amazon S3 storage

POST /api/v1/sign-s3

Headers

```
x-access-token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJvcmdhbml6YXRpb25JZCI6Im9yZzU0MzIxMTIzNDU2Nzg5M
Content-Type: application/json
Cache-Control: no-cache
```

Body

```
{
    "activity_id": "4",
    "activity_date": "2019-05-18T01:01:01Z",
    "activity_user_data": "some data",
    "file_name":"IMG_20181112_152423.jpg",
    "file_type":"image/jpg"
}
```

activity_id - visit identifier on the mobile application;

activity_date - date of the visit;

file_name - file name in the storage (e.g. GUID.png)

file_type - file format (e.g. image/png)

Result

```
{
    "signedRequest": "https://ctvision.s3.eu-central-1.amazonaws.com/Asset/6057e6e0-77c8-11e9-b380-220
    "url":"https://s3.amazonaws.com/ctvision/Asset/6057e6e0-77c8-11e9-b380-22000b12c2c9/text.txt"
}
```

As a response a JSON with 2 parameters will be received:

signedRequest - file download link;

url - file access link.

File upload to the S3 storage

PUT /{signedRequest}

For uploading PUT request will be used from the link obtained in the previous step. For this step, the use of the JWT is not required, since the link for Amazon has its own signature and active lifetime. This time is specified in the "X-Amz-Expires" parameter in seconds.

For the testing this time is 600 seconds, then it will be reduced to 60 seconds.

Headers

```
Content-Type: application/x-www-form-urlencoded
```

Body

```
binary file
```

Result

```
200 OK
```

Beginning of the recognition process

POST /api/v1/asset

After the image has been successfully uploaded into S3 storage, the recognition process can be started.

The current process means getting the results straight away.

Headers

```
x-access-token:eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJvcmdhbml6YXRpb25JZCI6Im9yZzU0MzIxMTIzNDU2Nzg5MC
Content-Type:application/json
```

Body

```
{
    "activity_id": "4",
    "activity_user_data": "some activity data",
    "user_data": "some asset data",
    "asset_id": "IMG_20181112_152423",
    "file_url": "https://s3.amazonaws.com/ctvision/Assets/6057e6e0-77c8-11e9-b380-22000b12c2c9/4c50265
    "ir_type": "type1"
}
```

activity_id - visit identifier on the mobile application;

asset_id - photo identifier;

activity_user_data - additional visit data (considered only at the first request, when a visit is created);

user_data - additional photo data'

file_url - access link to the uploaded to the S3 storage file;

ir_type - recognition algorithm type (available values: type1).

Result

```
{
    "asset": {
        "created_at": "2019-05-20T16:09:20.184Z",
        "uid": "a0182ec6-7b19-11e9-a77f-22000b12c2c9",
        "activity_uid": "4c502652-7b07-11e9-becd-22000b12c2c9",
        "is_wrong": null,
        "is_done": null,
        "file_url": "https://s3.amazonaws.com/ctvision/Assets/6057e6e0-77c8-11e9-b380-22000b12c2c9/4c
        "sfdc_id": "IMG_20181112_152423",
        "user_data": "some data"
    },
    "predictresponse": {
```

```
x-access-token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJvcmdhbml6YXRpb25JZCI6Im9yZzU0MzIxMTIzNDU2Nzg5M
Content-Type: application/json
Cache-Control: no-cache
```

Body

```
{
    "activity_id": "my_test_activity",
    "activity_date": null,
    "activity_user_data": "some activity data",
    "user_data": "some asset data",
    "file_name": "my_test_image2.png",
    "file_type": "image/png",
    "file_content": "file in base64 format"
}
```

Result

```
200 OK
```

After the request is fulfilled, the file will be uploaded to S3 and queued for recognition.

Validation of the processing results in the queue

GET /api/v1/checkAsset

For the validation of the recognition results of the queued file the use a GET request.

Headers

```
x-access-token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJvcmdhbml6YXRpb25JZCI6Im9yZzU0MzIxMTIzNDU2Nzg5M
Content-Type: application/json
Cache-Control: no-cache
```

Body

```
{
    "asset_uid": "UID of the asset"
}
```

Result, 200 OK

```
{
    "created_at": "2019-11-13T15:42:00.455Z",
    "asset_uid": "d417d43a-fbcd-11e9-8225-0685d8bbffd8",
    "result": {},
    "status": "success",
    "error": null,
    "model_id": "WTTUDAE76FSRK4Q3ZDD2OVXL7I"
}
```

result - contains the result with the accounted shelves;

status - processing result (success/error);

error - error message;

model_id - Einstein model used in processing.

If the file has not been processed yet, an error message will be displayed.

Obtaining the information on uploaded photos of the visit

POST /api/v1/activityAssets

This request allows getting the download links of the photos on the mobile device and the 'uid' of all photos for further obtaining the shelves results.